

In the Claims:

Please amend claim 29, as shown below.

1. (Previously Presented): A system for high availability clustering of a group of computer nodes, comprising:

A Java-based cluster server that allows an application to access a set of resources of various resource types, including application servers and transaction processing systems, within a cluster, wherein said resources, application servers, and transaction processing systems are available at one or more computers in the cluster, and wherein the resources, application servers, and transaction processing systems can be grouped by resource type within a pool of resources;

a resource interface provided by said Java-based cluster server that provides an abstraction layer and allows the Java-based cluster server to receive uniform requests from the application and communicate the requests to said set of resources;

a plurality of plugins that are plugged into the resource interface to provide a set of application-specific callbacks from the Java-based cluster server to the set of resources, wherein the system includes a plugin for each resource type, and wherein each plugin implements a resource API to encapsulate its particular resource type-specific behavior and to isolate the Java-based cluster server from that behavior while providing access to its pool of resources;

wherein a JNDI interface provides an interface between the Java-based cluster server and a JNDI-compliant database;

wherein additional plugins may be plugged into the resource interface for other resource types;

wherein the plurality of plugins include a plugin for an application server and a plugin for a transaction processing system; and

wherein the system can be extended by adding additional computers with Java-based cluster servers and resource interfaces operating thereon.

2. (Previously Presented): The system of claim 1 wherein each of said Java-based cluster servers includes a heartbeat interface that provides heartbeat information to other Java-based cluster servers at said other application servers.

3 – 4. (Canceled)

5. (Previously Presented): The system of claim 1 wherein the system includes a cluster administration utility for accessing and administering the Java-based cluster server using remote method invocation calls.

6. (Original): The system of claim 1 wherein each resource has a resource type associated with it.

7. (Original): The system of claim 6 wherein resources are the object instances of their respective resource types.

8. (Original): The system of claim 1 wherein a resource is any of a computer, internet protocol address, disk, database, or file system or application.

9. (Previously Presented): The system of claim 1 wherein the Java-based cluster server defines resource groups that includes clusters of resources.

10 - 11. (Canceled)

12. (Previously Presented): A method for providing a high availability clustering framework system for a group of computer nodes, comprising the steps of:

allowing an software application to access, via a computer and a Java-based cluster server operating thereon, a set of resources of various resource types, including application servers and transaction processing systems, within a Java-based cluster wherein said resources are available at said computer or at another computer, and wherein the resources and application servers can be grouped by resource type within a pool of resources;

providing a resource interface at said Java-based cluster server that provides an abstraction layer and allows the Java-based cluster server to receive uniform requests from the Java application and communicate the requests to said set of resources via a plurality of plugins that are plugged into the resource interface;

wherein the plurality of plugins are plugged into the resource interface to provide a set of application-specific callbacks from the Java-based cluster server to the set of resources, wherein the system includes a plugin for each resource type, and wherein each plugin implements a

resource API to encapsulate its particular resource type-specific behavior and to isolate the Java-based cluster server from that behavior while providing access to its pool of resources;

wherein a JNDI interface provides an interface between the Java-based cluster server and a JNDI-compliant database;

wherein additional plugins may be included in the resource interface for other resource types;

wherein the plurality of plugins include a plugin for an application server and a plugin for a transaction processing system; and

wherein the system can be extended by adding additional computers with Java-based cluster servers and resource interfaces operating thereon.

13. (Previously Presented): The method of claim 12 wherein said Java-based cluster server includes a heartbeat interface provides heartbeat information to other Java-based cluster servers at said other application servers.

14 - 15. (Canceled)

16. (Previously Presented): The method of claim 12 wherein the system includes a cluster administration utility for accessing and administering the Java-based cluster server using remote method invocation calls.

17. (Original): The method of claim 12 wherein each resources has a resource type associated with it.

18. (Original): The method of claim 17 wherein resources are the object instances of their respective resource types.

19. (Original): The method of claim 12 wherein a resource is any of a computer, ip address, disk, database, or file system or application.

20. (Previously Presented): The method of claim 12 wherein the Java-based cluster server allows for clustering resources within a resource group.

21 - 28. (Canceled)

29. (Currently Amended): A method for high-availability clustering, comprising the steps of:

receiving requests at a Java-based cluster server from [[a]] an application to access one or a plurality of application servers of different types within a cluster, wherein the application servers are available at one or more computers within the cluster;

communicating the requests to a Java-based cluster server that operates at the computer and provides access to the plurality of application servers, wherein the Java-based cluster server further comprises a resource interface that provides an abstraction layer and allows the Java-

based cluster server to receive uniform requests from the client application and communicate the requests to the application servers; and

using a plurality of plugins that can be plugged into the resource interface to provide application-specific callbacks from the Java-based cluster server to the application servers, wherein the system includes a plugin for each application server type, and wherein each plugin implements a resource interface that encapsulates the particular resource type-specific behavior for that application server type, and isolates the cluster server from that behavior while still providing access to the application server; and

wherein the plurality of plugins include a plugin for an application server and a plugin for a transaction processing system.

30. (Previously Presented): The method of claim 29 wherein, for each application server type, an appropriate plug-in is loaded at the time the first application server of a defined type is created, and wherein a handle is created to the specific resource instance, which can then be used by the Java-based cluster server in subsequent method calls.